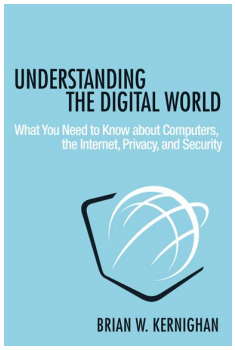
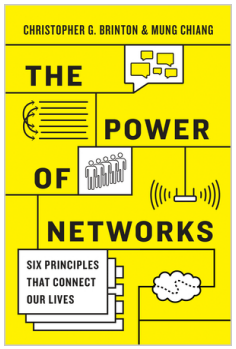
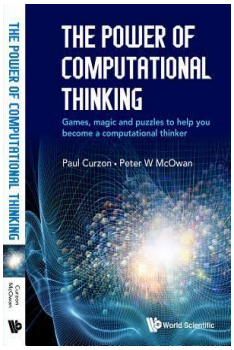
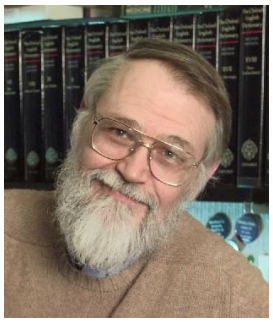
Understanding the Digital World, *Brian W. Kernighan*, Princeton University Press, (2017) ISBN 978-0691176543 (hbk), 256 p.
The Power of Networks, *Christopher G. Brinton, Mung Chiang*, Princeton University Press, (2017) ISBN 978-0691170718 (hbk), 328 p.
The Power of Computational Thinking, *Paul Curzon, Peter W. McOwan*, World Scientific, (2017) ISBN 978-1786341839 (hbk), 232 p.







Computers play an increasingly important role in mathematics and the converse is also true, old and new branches of mathematics are increasingly important in computer science. Traditionally, languages, science, and mathematics were the corner stones and the main tools in education to develop skills needed to understand the world. The authors of the third book however argue that some computational or algorithmic thinking is essentially different from the traditional skills and therefore it should be part of our educational system. But before one can formulate an opinion about whether or not this argument is justified, we should know what this computational thinking really means.


Brian Kernighan

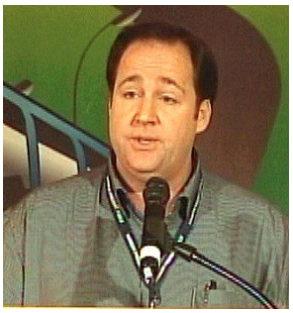
In my opinion, a good starting point is to acquire a vocabulary. ICT people talk a special language and not everybody is familiar with all of these terms. Brian Kernighan in his book discusses successively hardware, software, and communication. Basically it is the content of a course he has been teaching in Princeton. He wrote this book while turning 77, so he lived through the whole evolution since he started his career in Bell Labs late 1960s where he worked alongside K. Thompson and D. Ritchie, the creators of Unix and he was a pioneer of computer science himself. In this book he gives a modern account of all the terms that we all have heard mentioning, but which we may not be able to define precisely. We probably know what gigabyte, CPU, RAM, flash-drive, etc. means, but might have a problem to distinguish the World Wide Web from the Internet or who can explain the difference between ADSL, ethernet, Wi-Fi, and Bluetooth, or between DHPC, and TCP/IP, how exactly does cloud computing work, and how and by whom is our privacy threatened? The answers to these questions, and discussion of many many more terms, procedures, and technologies, all worth knowing, you can find in this very readable and entertaining book. It is very up-to-date including Snowden, recent privacy lawsuits, etc., a laudable attempt, as things are changing extremely fast in this area.

Of course, as a mathematician, perhaps you only use a computer as a tool to do simulations, to produce graphs, or just to typeset your papers, but privately, you will use a smartphone, do bank transactions, read an e-book on a tablet, watch television, and in a not too distant future we shall all be confronted with an Internet of Things. So even, if you consider yourself to be a die-hard pure mathematician, you have to deal with some or all of these devices that all have processing capacity and they are all more powerful than the IBM 1620 which I used to prepare my master thesis. If on the other hand, as a mathematician you have a more applied inclination, I know there are very challenging and even quite abstract problems to be found in computer science. In that case, you better learn the IT slang to enable communication. Both mathematics and computer science will profit from collaboration. It is my firm belief that most of the minor steps forward are obtained by digging deep into your mathematical sub-sub-sub problem, but the major progress will come from mixing different disciplines, if not within the vast body of mathematics itself, then from looking across the boundaries and explore problems from engineering, biotechnology, or whatever

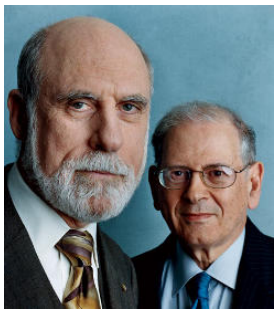
is appealing to you. Aren't most if not all sciences eventually applications of mathematics? The major difficulty to initiate such a collaboration is often the vocabulary. Not that Kernighan's book will start you doing computer science but it is the abc that allows you to start.

The book on *Networks* by Brinton and Chiang is somewhat similar but more restrictive in scope. The authors elaborate mostly on what falls in Kernighan's under the part on communication. Again, there is no mathematical background, although there is a lot of linear algebra, graph theory, optimization on graphs, data mining, machine learning, etc. involved. Think of Google's PageRank (an eigenvalue problem now worth billions of dollars), recommendation systems of Amazon, Netflix or YouTube, and the likes of it. Some of the material presented is 'technical' in the sense that it explains how this massive stream of data is organized or congestion is prevented, but also how these companies have organized their recommendation system, how they get paid for their services and how they compute a price for advertisement. How come that some items go 'viral' on the web, and how Facebook and Twitter can influence people, and how MOOCs (Massive Open Online Courses) are set up, and so on. Do not forget that computers are not only used for research, but they play an increasing role in our teaching (of mathematics and other courses). It requires some reflection on whether it is a good evolution or not to replace a proof of a theorem on the black- (or white-) board by some PowerPoint or another type of electronic projection.

Dennis Strigl



Eric Smidt



Vint Cerf, Robert Kahn

These authors also include some history (somewhat more than Kernighan) and they added four interviews with people who made all this happen: Dennis Strigl (former COO of Verizon, a wireless communication provider in the US), Eric Smidt (former CEO of Google), and Robert Kahn and Vint Cerf (both considered fathers of the Internet).

As mentioned in the beginning, the third book is about (the teaching of) computational thinking. They do not give (deep) arguments in any direction. In fact I believe they are already convinced that it should be included at an elementary level and they just give examples of what they believe it is to acquire this basic skill. It is somewhat surprising to read that in their opinion it is not learning to think as a computer does (if it can think at all), but it is how humans function in daily life and how this can be transferred to a machine. It is thus more than just the algorithmic idea, but also how a problem has to be modeled, based on scientific arguments, sometimes involving heuristics, and a great deal of pattern matching. The way in which the problem is modeled can allow for abstraction and generalization, or help to decompose the problem in smaller subproblems. And, not the least important, it also involves proper understanding of the problem that is communicated to you and knowing how to communicate the answer.

The authors come to this conclusion after they have illustrated these aspects with examples from daily life and by puzzles that they solve. Their arguments are not really systematic or scientifically underscored. Neither do they give recommendations on how their ideas should be implemented. In fact, what is in this book is basically a summary of what they published in the online magazine *Computer Science for Fun* located at `www.cs4fn.org` which is mainly a blog of the authors.

Adhemar Bultheel